Ein Bild, das Text, ClipArt enthält.

Automatisch generierte Beschreibung

Content

[1 Installation: 2](#_Toc110329475)

[1.1 In Eclipse: 2](#_Toc110329476)

[1.2 Command Line: 2](#_Toc110329477)

[2 Components: 2](#_Toc110329478)

[2.1 Problem: 2](#_Toc110329479)

[2.2 Method: 2](#_Toc110329480)

[2.3 Representation: 2](#_Toc110329481)

[2.4 Ranking: 3](#_Toc110329482)

[3 Information about the GUI 3](#_Toc110329483)

[4 Information about the command line 3](#_Toc110329484)

[5 Information about specific Components: 3](#_Toc110329485)

**Info: Everything was tested one Eclipse IDE version 2019-12 (4.14.0) and on Java Version 14.  
 Apart from the Installation only the Attachment-Version was used for Testing.**

# Installation:

## In Eclipse:

Unzip folder and import in Eclipse as existing project.

Open the project-explorer and go to src/main/FrevoMain, right click -> Run As -> Java Application.

## Command Line:

Download and unzip the project.

To start over the command line got to the projects root folder, then type “java -jar createscripts.jar” to build the necessary files. Then type “launch\_Frevo.bat” on Windows or “./launch\_Frevo.sh” on Linux, to start Frevo.  
For more command line options type type “launch\_Frevo.bat /?” on Windows or “./launch\_Frevo.sh /?” on Linux.

# Components:

Components are custom elements added into the program, that define a task to be optimized as well as how it should be optimized.  
Each component consists of an xml file defining its basic attributes as well as source code. The source code must include a Class inheriting from the corresponding component type’s base class, both of which are found in the Components folder.  
The AbstractComponent class is the class, that all different base classes for the components are inherited of.

## Problem:

Defines a task to optimize.

Base Class: AbstractSingleProblem for single problems, where only one agent is dropped in an environment and AbstractMultiproblem for multi problems, where multiple agents are dropped into an environment and have compete against each other. Both inherit from AbstractProblem.

## Method:

Defines how the neural networks evolve based on their fitness levels.

Base Class: AbstractMethod

## Representation:

Defines the way an agent is represented as a neural network.

Base Class: AbstractRepresentation

Extensions:   
AbstractEvolutionStatus: information about the representation for the optimise Method, currently only implemented as Part of NNGA.

## Ranking:

Defines how the fitness levels of the agents are determined.

Base Class: AbstractRanking

# Information about the GUI

After changing, the problem you might have to select a new Ranking system as well before starting the simulation.

Connecting to an external simulator is not implemented yet.

# Information about the command line

Executing a results File from the command line only works with single problems.

# Information about specific Components:

**Problems:**

Tic Tac Toe 2x2 has no replay.

On Linux running EmergencyExit requires adapting the Filename parameter.

**Methods:**

PSO requires a single Problem.

PSO is only compatible with fully meshed net.

CuckooSearch requires a single Problem.

NoveltySearch requires a single Problem.

NoveltySpecies requires a single Problem.

GASpecies requires a single Problem.

No Problem implements Novelty ranking, making Novelty ranking, Novelty search and Novelty species unusable.

CuckooSearch requires either FullyMeshedNet or Fully connected neural network.

**Representations:**

MealyFSM: min\_input\_values, max\_input\_values, treshhold\_for\_inputs, threshold\_units, and unit\_input\_values require exactly as many numbers as there are different inputs.

CAM is the only problem, that currently support bulk representations.

**Rankings:**

MultiSort is currently incompatible with all methods except RandomSearch and NNGA.  
Incompatibility with PSO, CuckooSearch, NoveltySearch, NoveltySpecies and GASpecies due to those methods requiring a single Problem and MultiSort requiring a multi problem.  
Incompatibility with CEA2D is due to CEA2D always comparing 9 different networks, and Multi Sort needing a multiple of the number of competing networks in one game.

Swiss System only works for 1vs1 problems, for more than that use MultiSwiss System.